

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

Compressed Timing Indicators for Media Samples

Inventor:

Gary J. Sullivan

ATTORNEY'S DOCKET NO. MS1-946US

1 **RELATED APPLICATIONS**

2 This application claims the benefit of U.S. Provisional Application No.
3 60/241,407 filed October 18, 2000, the disclosure of which is incorporated by
4 reference herein.

5
6 **TECHNICAL FIELD**

7 The present invention relates to video processing systems and, more
8 particularly, to the compression of timing indicators associated with media
9 samples.

10
11 **BACKGROUND**

12 The concept of recording and using timing information is fundamental to
13 the needs of multimedia applications. Pictures, video, text, graphics, and sound
14 need to be recorded with some understanding of the time associated with each
15 sample of the media stream. This is useful for synchronizing different multimedia
16 streams with each other, for carrying information to preserve the original timing of
17 the media when playing a media stream, for identifying specific locations within a
18 media stream, and for recording the time associated with the media samples to
19 create a scientific or historical record. For example, if audio and video are
20 recorded together but handled as separate streams of media data, then timing
21 information is necessary to coordinate the synchronization of these two (or more)
22 streams.

23 Typically, a media stream (such as a recorded audio track or recorded video
24 or film shot) is represented as a sequence of media samples, each of which is
25 associated (implicitly or explicitly) with timing information. A good example of

1 this is video and motion picture film recording, which is typically created as a
2 sequence of pictures, or frames, each of which represents the camera view for a
3 particular short interval of time (e.g., typically 1/24 seconds for each frame of
4 motion picture film). When this sequence of pictures is played back at the same
5 number of frames per second (known as the "frame rate") as used in the recording
6 process, an illusion of natural movement of the objects depicted in the scene can
7 be created for the viewer.

8 Similarly, sound is often recorded by regularly sampling an audio
9 waveform to create a sequence of digital samples (for example, using 48,000
10 samples per second) and grouping sets of these samples into processing units
11 called frames (e.g., 64 samples per frame) for further processing such as digital
12 compression encoding or packet-network transmission (such as Internet
13 transmission). A receiver of the audio data will then reassemble the frames of
14 audio that it has received, decode them, and convert the resulting sequence of
15 digital samples back into sound using electro-acoustic technology.

16 Fig. 1 illustrates a conventional system 100 for processing and distributing
17 video content. The video content is captured using a video camera 102 (or any
18 other video capture device) that transfers the captured video content onto video
19 tape or another storage medium. Later, the captured video content may be edited
20 using a video editor 104. A video encoder 106 encodes the video content to
21 reduce the storage space required for the video content or to reduce the
22 transmission bandwidth required to transmit the video content. Various encoding
23 techniques may be used to compress the video content, such as the MPEG-2
24 (Moving Picture Experts Group 2nd generation) compression format.
25

1 In each of the above cases, a sample or group of samples in time of a media
2 stream can be identified as a frame, or fundamental processing unit. If a frame
3 consists of more than one sample in time, then a convention can be established in
4 which the timing information represented for a frame corresponds to the time of
5 some reference point in the frame such as the time of the first, last or middle
6 sample.

7 In some cases, a frame can be further subdivided into even smaller
8 processing units, which can be called fields. One example of this is in the use of
9 interlaced-scan video, in which the sampling of alternating lines in a picture are
10 separated so that half of the lines of each picture are sampled as one field at one
11 instant in time, and the other half of the lines of the picture are then sampled as a
12 second field a short time later. For example, lines 1, 3, 5, etc. may be sampled as
13 one field of picture, and then lines 0, 2, 4, etc. of the picture may be sampled as
14 the second field a short time later (for example 1/50th of a second later). In such
15 interlaced-scan video, each frame can be typically separated into two fields.

16 Similarly, one could view a grouping of 64 samples of an audio waveform
17 for purposes of data compression or packet-network transmission to be a frame,
18 and each group of eight samples within that frame to be a field. In this example,
19 there would be eight fields in each frame, each containing eight samples.

20 In some methods of using sampled media streams that are well known in
21 the art, frames or fields may consist of overlapping sets of samples or
22 transformations of overlapping sets of samples. Two examples of this behavior
23 are the use of lapped orthogonal transforms [1) Henrique Sarmiento Malvar, *Signal*
24 *Processing with Lapped Transforms*, Boston, MA, Artech House, 1992; 2) H. S.
25 Malvar and D. H. Staelin, "The LOT: transform coding without blocking effects,"

1 IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 37, pp. 553-
2 559, Apr. 1989; 3) H. S. Malvar, Method and system for adapting a digitized
3 signal processing system for block processing with minimal blocking artifacts,
4 U.S. Patent No. 4,754,492, June 1988.] and audio redundancy coding [1) J. C.
5 Bolot, H. Crepin, A. Vega-Garcia: "Analysis of Audio Packet Loss in the
6 Internet", Proceedings of the 5th International Workshop on Network and
7 Operating System Support for Digital Audio and Video, pp. 163-174, Durham,
8 April 1995; 2) C. Perkins, I. Kouvelas, O. Hodson, V. Hardman, M. Handley, J. C.
9 Bolot, A. Vega-Garcia, S. Fosse-Parisis: "RTP Payload for Redundant Audio
10 Data", Internet Engineering Task Force Request for Comments RFC2198, 1997.].
11 Even in such cases it is still possible to establish a convention by which a time is
12 associated with a frame or field of samples.

13 In some cases, the sampling pattern will be very regular in time, such as in
14 typical audio processing in which all samples are created at rigidly-stepped times
15 controlled by a precise clock signal. In other cases, however, the time between
16 adjacent samples in a sequence may differ from location to location in the
17 sequence.

18 One example of such behavior is when sending audio over a packet
19 network with packet losses, which may result in some frames not being received
20 by the decoder while other frames should be played for use with their original
21 relative timing. Another example of such behavior is in low-bit-rate
22 videoconferencing, in which the number of frames sent per second is often varied
23 depending on the amount of motion in the scene (since small changes take less
24 data to send than large changes, and the overall channel data rate in bits per second
25 is normally fixed).

1 If the underlying sampling structure is such that there is understood to be a
2 basic frame or field processing unit sampling rate (although some processing units
3 may be skipped), then it is useful to be able to identify a processing unit as a
4 distinct counting unit in the time representation. If this is incorporated into the
5 design, the occurrence of a skipped processing unit may be recognized by a
6 missing value of the counting unit (e.g., if the processing unit count proceeds as 1,
7 2, 3, 4, 6, 7, 8, 9, ..., then it is apparent that count number 5 is missing).

8 If the underlying sampling structure is such that the sampling is so irregular
9 that there is no basic processing unit sampling rate, then what is needed is simply a
10 good representation of true time for each processing unit. Normally however, in
11 such a case there should at least be a common time clock against which the
12 location of the processing unit can be referenced.

13 In either case (with regular or irregular sampling times), it is useful for a
14 multimedia system to record and use timing information for the samples or frames
15 or fields of each processing unit of the media content.

16 Different types of media may require different sampling rates. If timing
17 information is always stored with the same precision, a certain amount of rounding
18 error may be introduced by the method used for representing time. It is desirable
19 for the recorded time associated with each sample to be represented precisely in
20 the system with little or no such rounding error. For example, if a media stream
21 operates at 30,000/1001 frames per second (the typical frame rate of North
22 American standard NTSC broadcast video – approximately 29.97 frames per
23 second) and the precision of the time values used in the system is to one part in
24 10^{-6} seconds, then although the time values may be very precise in human terms, it
25 may appear to processing elements within the system that the precisely-regular

sample timing (e.g. 1001/30,000 seconds per sample) is not precisely regular (e.g. 33,366 clock increment counts between samples, followed by 33,367 increments, then 33,367 increments, and then 33,366 increments again). This can cause difficulties in determining how to properly handle the media samples in the system.

Another problem in finding a method to represent time is that the representation may “drift” with respect to true time as would be measured by a perfectly ideal “wall clock”. For example, if the system uses a precisely-regular sample timing of 1001/30,000 seconds per sample and all samples are represented with incremental time intervals being 33,367 increments between samples, the overall time used for a long sequence of such samples will be somewhat longer than the true time interval – a total of about one frame time per day and accumulating more than five minutes of error after a year of duration.

Thus, “drift” is defined as any error in a timecode representation of sampling times that would (if uncorrected) tend to increase in magnitude as the sequence of samples progresses.

One example of a method of representing timing information is found in the SMPTE 12M design [Society of Motion Picture and Television Engineers, Recommended Practice 12M : 1999] (hereinafter called “SMPTE timecode”). SMPTE timecodes are typically used for television video data with timing specified in the United States by the National Television Standards Committee (NTSC) television transmission format, or in Europe, by the Phase Alternating Line (PAL) television transmission format.

SMPTE timecode is a synchronization signaling method originally developed for use in the television and motion picture industry to deal with video

1 In other applications, namely video, the equipment uses timecode internally
2 to synchronize multiple timecode sources into one. An example would be a video
3 editor that synchronizes with timecode from a number of prerecorded scenes. As
4 each scene is combined with the others to make the final product, their respective
5 timecodes are synchronized with new timecode being recorded to the final
6 product.

7 SMPTE timecode provides a unique address for each frame of a video
8 signal. This address is an eight digit number, based on the 24 hour clock and the
9 video frame rate, representing Hours, Minutes, Seconds and Frames in the
10 following format:

11 **HH:MM:SS:FF**

12 The values of these fields range from 00 to 23 for HH, 00 to 59 for MM, 00
13 to 59 for SS, and 00 to 24 or 29 for FF (where 24 is the maximum for PAL 25
14 frame per second video and 29 is the maximum for NTSC 30,000/1001 frame per
15 second video). By convention, the first frame of a day is considered to be marked
16 as 00:00:00:01 and the last is 00:00:00:00 (one frame past the frame marked
17 23:59:59:24 for PAL and 23:59:59:29 for NTSC). This format represents a
18 nominal clock time, the nominal duration of scene or program material and makes
19 approximate time calculations easy and direct.

20 The frame is the smallest unit of measure within SMPTE timecode and is a
21 direct reference to the individual "picture" of film or video. The frame rate is the
22 number of times per second that pictures are displayed to provide a rendition of
23 motion. There are two standard frame rates (frames/sec) that typically use
24 SMPTE timecode: 25 frames per second and 30,000/1001 frames per second
25 (approximately 29.97 frames per second). The 25 frame per second rate is based

1 measured by a true "wall clock" will accumulate more than 86 seconds of error per
2 day. When using SMPTE drop frame counting, the drift accumulation magnitude
3 can be reduced by about a factor of about 1,000 (although the drift is still not
4 entirely eliminated and the remaining drift is still more than two frame sampling
5 periods).

6 The SMPTE timecode has been widely used in the video production
7 industry (for example, it is incorporated into the design of many video tape
8 recorders). It is therefore very useful if any general media timecode design is
9 maximally compatible with this SMPTE timecode. If such compatibility can be
10 achieved, this will enable equipment designed for the media timecode to work
11 well with other equipment designed specifically to use the SMPTE timecode.

12 Within this document, the following terminology is used. A timecode
13 describes the data used for representing the time associated with a media sample,
14 frame, or field. It is useful to separate the data of a timecode into two distinct
15 types: the timebase and the timestamp. The timestamp includes the information
16 that is used to represent the timing for a specific processing unit (a sample, frame,
17 or field). The timebase contains the information that establishes the basis of the
18 measurements units used in the timestamp. In other words, the timebase is the
19 information necessary to properly interpret the timestamps. The timebase for a
20 media stream normally remains the same for the entire sequence of samples, or at
21 least for a very large set of samples.

22 For example, we may interpret the SMPTE timecode as having a timebase
23 that consists of:

- 24 • Knowledge of (or an indication of) whether the system is NTSC or PAL,
25 and

- 1 • Knowledge of (or an indication of) whether or not the system uses SMPTE
2 "drop frame" counting in order to partially compensate for drift.

3
4 Given this, the timestamps then consist of the representations of the
5 parameters Hours, Minutes, Seconds, and Frames for each particular video frame.

6 Many existing systems transmit all parameters of the timestamp with each
7 frame. Since many of the parameters (e.g., hours and minutes) do not typically
8 change from one frame to the next, transmitting all parameters of the timestamp
9 with each frame results in the transmission of a significant amount of redundant
10 data. This transmission of redundant data results in the transmission of more data
11 than is necessary to communicate the current timing information.

12 The systems and methods described herein provide for the communication
13 of timing indicators that convey timing information using a reduced amount of
14 data.

15
16
17 **SUMMARY**

18 The systems and methods described herein provide for two different types
19 of timestamps to be transmitted along with frames of data. A full timestamp
20 includes complete timing information, such as hour information, minute
21 information, second information, and a frame number. A compressed timestamp
22 includes a portion of the complete timing information, such as the frame number.
23 When a receiving device receives a compressed timestamp, the receiving device
24 maintains the previous values of the timing parameters that are not contained in
25 the compressed timestamp. Since the most of the information in a full timestamp

is redundant from one frame to the next, sending a significant number of compressed timestamps between full timestamps reduces the amount of data that is transmitted, but does not result in a loss of timing information.

In one embodiment, a first frame of data is encoded. A first timestamp is generated and associated with the first frame of data. The first timestamp includes complete timing information. The first frame of data and the associated first timestamp is then transmitted to a destination. A second frame of data is encoded and a second timestamp associated with the second frame of data is generated. The second timestamp includes a portion of the complete timing information. The second frame of data and the associated second timestamp is transmitted to the destination.

In another embodiment, multimedia content to be encoded is identified. The identified multimedia content is encoded into multiple frames of data. Full timestamps are generated and associated with a portion of the frames of data. Each full timestamp contains complete time information. Compressed timestamps are generated and associated with frames of data that are not associated with a full timestamp. Each compressed timestamp contains a portion of the complete time information.

In a described embodiment, the full timestamps include hour information, minute information, second information, and a frame number.

In a particular implementation, the compressed timestamps include a frame number.

1 **BRIEF DESCRIPTION OF THE DRAWINGS**

2 Fig. 1 illustrates a conventional system for processing and distributing
3 video content.

4 Fig. 2 illustrates an example multimedia encoding system and an example
5 multimedia decoding system.

6 Fig. 3 is a flow diagram illustrating a procedure for encoding multimedia
7 content and transmitting timestamps and associated multimedia content frames.

8 Fig. 4 is a flow diagram illustrating a procedure for decoding multimedia
9 content that includes multiple time stamps and associated content frames.

10 Fig. 5 illustrates an example of a suitable operating environment in which
11 the systems and methods described herein may be implemented.

12
13
14
15 **DETAILED DESCRIPTION**

16 The systems and methods described herein utilize different types of timing
17 indicators (referred to as timestamps) to communicate timing information along
18 with frames of data. The use of both full timestamps and compressed timestamps
19 reduces the amount of timing information that must be communicated with the
20 frames of data. A full timestamp includes all timing information and is sent
21 occasionally (e.g., a few times each second or once every X frames of data).
22 Between full timestamps, a series of compressed timestamps are communicated
23 with the frames of data. The compressed timestamps contain a subset of the
24 complete timing information contained in the full timestamps. The compressed
25

1 timestamp contains the timing information that has changed since the last full
2 timestamp was sent.

3 Fig. 2 illustrates an example multimedia encoding system and an example
4 multimedia decoding system. A multimedia content source 202 provides
5 multimedia content (e.g., audio content, video content, or combined audio and
6 video content) to an encoder 204. Multimedia content source may be, for
7 example, a video camera, microphone or other capture device, or a storage device
8 that stores previously captured multimedia content. Encoder 204 includes a clock
9 206 and a frame counter 208. Clock 206 is used to determine timestamp
10 information and synchronize operation of encoder 204. Frame counter 208 keeps
11 track of consecutive frame numbers associated with frames of data. Encoder 204
12 also includes an encoding engine 210, which encodes multimedia content and
13 other data (such as timestamp information) into multiple frames. The output of
14 encoder 204 is communicated to a transmitter 212, which transmits the encoded
15 content to one or more receivers. Alternatively, transmitter 212 may be a storage
16 device that stores the encoded content (e.g., on a DVD, magnetic tape, or other
17 storage device).

18 Receiver 220 receives an encoded signal including one or more frames and
19 communicates the received signal to a decoder 222. Alternatively, receiver 220
20 may be a device (such as a audio player and/or a video player) capable of reading
21 stored encoded content (e.g., stored on a DVD or other storage device). Decoder
22 222 includes a clock 224 and a counter 226. Clock 224 aids in synchronizing
23 decoder 222. Counter 226 is used to assign frame identifiers to received frames of
24 data. Decoder 222 also includes a decoding engine 228 which decodes the
25 received signal. After decoding the received signal, decoder 222 communicates

1 the decoded content to a multimedia player 230 which renders the multimedia
2 content defined by the decoded signal. Multimedia player may be an audio player
3 (e.g., a CD player), a video player (e.g., a DVD player), or a combination audio
4 player and video player. Decoder 222 may be a separate device or may be
5 incorporated into another device, such as a television or a DVD player.

6 Fig. 3 is a flow diagram illustrating a procedure 300 for encoding
7 multimedia content and transmitting timestamps and associated multimedia
8 content frames. Initially, procedure 300 identifies a number of basic units per
9 second in a reference clock (block 302), which is represented by a parameter
10 labeled "base_ups". In a particular example, the reference clock has 30,000 basic
11 units per second (also referred to as 30,000 hertz). The procedure then identifies a
12 number of basic units of the reference clock per media sample period (block 304),
13 which is represented by a parameter labeled "base_upp". In a particular example,
14 each increment of a counter (such as a frame counter) occurs after 1001
15 increments of the reference clock. In this example, if the counter advances by
16 five, the reference clock advances by 5005. This example reduces the amount of
17 data that needs to be communicated regarding the clock (i.e., sending "5" instead
18 of "5005").

19 The procedure 300 then identifies a counting type that defines the manner
20 in which samples (or frames) are counted (block 306). Additional details
21 regarding the various counting types are provided below. At block 308, the
22 base_ups, base_upp, and counting type data is transmitted to one or more
23 receivers. These data values allow each receiver to understand and properly
24 decode subsequent frames of data.
25

Next, the procedure receives multimedia content to be encoded and creates a first content frame (block 310). The first content frame is created by encoding a portion of the received multimedia content. The procedure then transmits a full timestamp along with the first content frame (block 312). The full timestamp may be embedded within the first content frame or transmitted separately, but along with the full timestamp. The full timestamp includes the hour, minutes, seconds, and frame number associated with the first content frame.

The procedure then creates the next content frame by encoding the next portion of the received multimedia content (block 314). At block 316, the procedure determines whether to transmit a full timestamp or a compressed timestamp. As mentioned above, a full timestamp includes all time-related information (i.e., the hour, minute, second, and frame number associated with the first content frame). The compressed timestamp includes a subset of the information required for a full timestamp. In a particular implementation, the compressed timestamp contains the information that has changed since the last timestamp (either full or compressed) was transmitted to the receivers. Typically, the compressed timestamp includes the frame number associated with the current content frame being transmitted. The compressed timestamp reduces the amount of data that must be transmitted when compared with the full timestamp. In a particular implementation, the full timestamp is sent several times each second. In an alternate implementation, the full timestamp is sent every X frames, where X is approximately 15.

In another implementation, the decision of whether to send a full timestamp or compressed timestamp is adjusted dynamically based on an estimate of the reliability of the communication link between transmitter and receiver. If the

1 estimated reliability of the communication link is high, then full timestamps may
2 be sent less frequently. However, if the communication link is not expected to be
3 reliable, the full timestamps are sent more frequently.

4 If the procedure determines that a full timestamp should be transmitted, a
5 full timestamp is transmitted along with the next content frame (block 318).
6 Otherwise, a compressed timestamp is transmitted along with the next content
7 frame (block 320). The procedure continues by returning to block 314 to create
8 the next content frame and determine whether a full timestamp or a compressed
9 timestamp is to be transmitted along with the next content frame.

10 In a particular embodiment, the data that specifies the timebase and the
11 starting timestamp of a sequence of data samples (or frames) is sent using the
12 following pseudo-code:

13 send (base_ups) // unsigned integer
14 send (base_upp) // unsigned integer
15 send (counting_type) // defined in Table 1
16 send (full_timestamp_sequence_flag) // boolean
17 send (discontinuity_flag) // boolean
18 send (count_dropped) // boolean
19 send (frames_value) // integer
20 if (counting_type != '000')
21 send (offset_value) // integer
22 send (seconds_value) // integer
23 send (minutes_value) // integer
24 send (hours_value) // integer

25 These data specify the time of the first sample of a sequence of frames and specify
the timebase necessary for interpretation of the parameters of each individual
timestamp. Since these data specify both the timebase and the initial timestamp
for an entire sequence of frames, they are referred to herein as the sequence header

information for this particular embodiment. In one embodiment, a full timestamp is included in each sequence header. Alternatively, the sequence headers may not contain a full timestamp. Instead, the data contained in a full timestamp is retrieved from the full timestamp associated with the first frame of data following the sequence header.

The `base_ups`, `base_upp`, and counting type parameters are discussed above. Table 1 below defines the various `counting_type` values.

TABLE 1

Value	Meaning
000	No dropping of frames_value count values and no use of offset_value
001	No dropping of frames_value count values
010	Dropping of individual zero values of frames_value count
011	Dropping of individual max_pps values of frames_value count
100	Dropping of the two lowest (values 0 and 1) frames_value counts when seconds_value is zero and minutes_value is not an integer multiple of ten
101	Dropping of unspecified individual frames_value count values
110	Dropping of unspecified numbers of unspecified frames_value count values
111	Reserved

Particular parameters are defined as follows:

- **full_timestamp_sequence_flag**: Indicates whether every timestamp in the following sequence of timestamps shall be fully specified or whether

some timestamps (referred to as compressed timestamps) may only contain partial information (depending on memory of values sent previously in the sequence header or in a frame timestamp). If full_timestamp_sequence flag is "1", then full_timestamp_flag must be "1" in the timestamp information for every frame in the following sequence.

- **discontinuity_flag:** Indicates whether the time difference that can be calculated between the starting time of the sequence and the time indicated for the last previous transmitted frame can be interpreted as a true time difference. Shall be "1" if no previous frame has been transmitted.
- **count_dropped:** Indicates, if discontinuity_flag is "0", whether some value of frames_value was skipped after the last previous transmitted frame to reduce drift between the time passage indicated in the seconds_value, minutes_value, and hours_value parameters and those of a true clock.
- **frames_value, offset_value, seconds_value, minutes_value, and hours_value:** Indicate the parameters to be used in calculating an equivalent timestamp for the first frame in the sequence. Shall be equal to the corresponding values of these parameters in the header of the first frame after the sequence header, if present in the sequence header.

In this embodiment, an extra signed-integer parameter called offset_value is used in addition to the unsigned integer frames_value, seconds_value, minutes_value, and hours_value parameters that are used by the SMPTE

timecode's timestamp, in order to relate the time of a sample precisely relative to true time, as shown in a formula below.

In a particular embodiment, the timestamp structure sending process for the timestamps on individual media samples (or frames) is implemented using the following pseudo-code:

```
send (full_timestamp_flag)      // boolean
send (frames_value)             // unsigned integer
if (counting_type != '000') {
    if (full_timestamp_flag)
        send (offset_value)      // signed integer
    else {
        send (offset_value_flag) // boolean
        if (offset_value_flag)
            send (offset_value)  // signed integer
    }
    if (counting_type != '001')
        send (count_dropped_flag) // boolean
}
if (full_timestamp_flag) {
    send (seconds_value)          // unsigned integer 0..59
    send (minutes_value)         // unsigned integer 0..59
    send (hours_value)           // unsigned integer
} else {
    send (seconds_flag)          // boolean
    if (seconds_flag) {
        send (seconds_value)     // unsigned integer 0..59
        send (minutes_flag)      // boolean
        if (minutes_flag) {
            send (minutes_value)  // unsigned integer 0..59
            send (hours_flag)     // boolean
            if (hours_flag)
                send (hours_value) // unsigned integer
        }
    }
}
```

1 If any timestamp is incomplete (i.e., full_timestamp_flag is zero and at least one of
2 seconds_flag, minutes_flag, hours_flag, and offset_value_flag is present and zero)
3 the last prior sent value for each missing parameter is used. An equivalent time
4 specifying the time of a media sample (in units of seconds) may be computed as
5 follows:

$$\text{equivalent_time} = 60 \times (60 \times \text{hours_value} + \text{minutes_value}) + \text{seconds_value} + \\ (\text{base_upp} \times \text{frames_value} + \text{offset_value}) / \text{base_ups}$$

6
7
8
9
10 Using the timebase parameters, a derived parameter is defined as:

$$\text{max_pps} = \text{ceil}(\text{base_ups} / \text{base_upp})$$

11
12
13 where ceil(x) is defined as the function of an argument x, which, for non-negative
14 values of x, is equal to x if x is an integer and is otherwise equal to the smallest
15 integer greater than x. The value of frames_value should not exceed max_pps.
16

17 If count_dropped_flag is '1', then:

18 if counting_type is '010', frames_value shall be '1' and the value of
19 frames_value for the last previous transmitted frame shall not
20 be equal to '0' unless a sequence header is present between
21 the two frames with discontinuity_flag equal to '1'.

22 if counting_type is '011', frames_value shall be '0' and the value of
23 frames_value for the last previous transmitted frame shall not
24 be equal to max_pps unless a sequence header is present
25 between the two frames with discontinuity_flag equal to '1'.

1 if counting_type is '100', frames_value shall be '2' and the
2 seconds_value shall be zero and minutes_value shall not be
3 an integer multiple of ten and frames_value for the last
4 previous transmitted frame shall not be equal to '0' or '1'
5 unless a sequence header is present between the two frames
6 with discontinuity_flag equal to '1'.

7 if counting_type is '101' or '110', frames_value shall not be equal to
8 one plus the value of frames_value for the last previous
9 transmitted frame modulo max_pps unless a sequence header
10 is present between the two frame with discontinuity_flag
11 equal to '1'.

12 As the degree of precision for the various parameters of each media sample
13 timestamp becomes coarser, the inclusion of the further information needed to
14 place the timestamp within the more global scale is optional. Any coarse-level
15 context information that is not sent is implied to have the same value as the last
16 transmitted parameter of the same type. The finely-detailed information necessary
17 to locate the precise sample time relative to that of neighboring samples is
18 included with every timestamp, but as the degree of coarseness of the time
19 specification becomes higher, the inclusion of further more coarse context
20 information is optional in order to reduce the average amount of information that
21 is required to be communicated.

22 Fig. 4 is a flow diagram illustrating a procedure 400 for decoding
23 multimedia content that includes multiple time stamps and associated content
24 frames. At block 402, the procedure receives base_ups, base_upp, and counting
25 type data associated with a multimedia stream from a transmitting device. This

1 information allows the receiving system to properly interpret and decode the
2 subsequently received content. Next, a full timestamp and an associated first
3 multimedia content frame are received (block 404). The full timestamp provides
4 the hours, minutes, seconds, and frame number associated with the first received
5 frame, and, in a particular embodiment, a time offset number allowing a drift-free
6 precise relation to be determined between the time computed from the other
7 parameters and the true time of the sample.

8 The procedure 400 then receives a next multimedia content frame and an
9 associated timestamp. An associated flag (`full_timestamp_flag`) will indicate
10 whether the timestamp is a full timestamp or a compressed timestamp. The
11 procedure decodes the multimedia content frame (block 408) and determines
12 (based on the `full_timestamp_flag`) whether the timestamp is a full timestamp or a
13 compressed timestamp (block 410). If the timestamp is a full timestamp, the
14 procedure updates all timing parameters provided by the full timestamp (block
15 412). If the timestamp is a compressed timestamp, the procedure updates the
16 frame parameter (block 414). The system uses the values from the most recent
17 full timestamp for all other timing parameter values. Alternatively, if a
18 compressed timestamp is received, the procedure updates all timing parameters
19 contained in the compressed timestamp.

20 After updating one or more timing parameters, the procedure returns to
21 block 406 to receive and process the next multimedia content frame and associated
22 timestamp.

23 Fig. 5 illustrates an example of a suitable computing environment 500
24 within which the video encoding and decoding procedures may be implemented
25

1 (either fully or partially). The computing environment 500 may be utilized in the
2 computer and network architectures described herein.

3 The exemplary computing environment 500 is only one example of a
4 computing environment and is not intended to suggest any limitation as to the
5 scope of use or functionality of the computer and network architectures. Neither
6 should the computing environment 500 be interpreted as having any dependency
7 or requirement relating to any one or combination of components illustrated in the
8 exemplary computing environment 500.

9 The video encoding and decoding systems and methods described herein
10 may be implemented with numerous other general purpose or special purpose
11 computing system environments or configurations. Examples of well known
12 computing systems, environments, and/or configurations that may be suitable for
13 use include, but are not limited to, personal computers, server computers,
14 multiprocessor systems, microprocessor-based systems, network PCs,
15 minicomputers, mainframe computers, distributed computing environments that
16 include any of the above systems or devices, and so on. Compact or subset
17 versions may also be implemented in clients of limited resources.

18 The computing environment 500 includes a general-purpose computing
19 device in the form of a computer 502. The components of computer 502 can
20 include, by are not limited to, one or more processors or processing units 504, a
21 system memory 506, and a system bus 508 that couples various system
22 components including the processor 504 to the system memory 506.

23 The system bus 508 represents one or more of several possible types of bus
24 structures, including a memory bus or memory controller, a peripheral bus, an
25 accelerated graphics port, and a processor or local bus using any of a variety of

1 bus architectures. By way of example, such architectures can include an Industry
2 Standard Architecture (ISA) bus, a Micro Channel Architecture (MCA) bus, an
3 Enhanced ISA (EISA) bus, a Video Electronics Standards Association (VESA)
4 local bus, and a Peripheral Component Interconnects (PCI) bus also known as a
5 Mezzanine bus.

6 Computer 502 typically includes a variety of computer readable media.
7 Such media can be any available media that is accessible by computer 502 and
8 includes both volatile and non-volatile media, removable and non-removable
9 media.

10 The system memory 506 includes computer readable media in the form of
11 volatile memory, such as random access memory (RAM) 510, and/or non-volatile
12 memory, such as read only memory (ROM) 512. A basic input/output system
13 (BIOS) 514, containing the basic routines that help to transfer information
14 between elements within computer 502, such as during start-up, is stored in ROM
15 512. RAM 510 typically contains data and/or program modules that are
16 immediately accessible to and/or presently operated on by the processing unit 504.

17 Computer 502 may also include other removable/non-removable,
18 volatile/non-volatile computer storage media. By way of example, Fig. 5
19 illustrates a hard disk drive 516 for reading from and writing to a non-removable,
20 non-volatile magnetic media (not shown), a magnetic disk drive 518 for reading
21 from and writing to a removable, non-volatile magnetic disk 520 (e.g., a "floppy
22 disk"), and an optical disk drive 522 for reading from and/or writing to a
23 removable, non-volatile optical disk 524 such as a CD-ROM, DVD-ROM, or other
24 optical media. The hard disk drive 516, magnetic disk drive 518, and optical disk
25 drive 522 are each connected to the system bus 508 by one or more data media

1 other input devices are connected to the processing unit 504 via input/output
2 interfaces 540 that are coupled to the system bus 508, but may be connected by
3 other interface and bus structures, such as a parallel port, game port, or a universal
4 serial bus (USB).

5 A monitor 542 or other type of display device can also be connected to the
6 system bus 508 via an interface, such as a video adapter 544. In addition to the
7 monitor 542, other output peripheral devices can include components such as
8 speakers (not shown) and a printer 546 which can be connected to computer 502
9 via the input/output interfaces 540.

10 Computer 502 can operate in a networked environment using logical
11 connections to one or more remote computers, such as a remote computing device
12 548. By way of example, the remote computing device 548 can be a personal
13 computer, portable computer, a server, a router, a network computer, a peer device
14 or other common network node, and so on. The remote computing device 548 is
15 illustrated as a portable computer that can include many or all of the elements and
16 features described herein relative to computer 502.

17 Logical connections between computer 502 and the remote computer 548
18 are depicted as a local area network (LAN) 550 and a general wide area network
19 (WAN) 552. Such networking environments are commonplace in offices,
20 enterprise-wide computer networks, intranets, and the Internet.

21 When implemented in a LAN networking environment, the computer 502 is
22 connected to a local network 550 via a network interface or adapter 554. When
23 implemented in a WAN networking environment, the computer 502 typically
24 includes a modem 556 or other means for establishing communications over the
25 wide network 552. The modem 556, which can be internal or external to computer

502, can be connected to the system bus 508 via the input/output interfaces 540 or other appropriate mechanisms. It is to be appreciated that the illustrated network connections are exemplary and that other means of establishing communication link(s) between the computers 502 and 548 can be employed.

In a networked environment, such as that illustrated with computing environment 500, program modules depicted relative to the computer 502, or portions thereof, may be stored in a remote memory storage device. By way of example, remote application programs 558 reside on a memory device of remote computer 548. For purposes of illustration, application programs and other executable program components such as the operating system are illustrated herein as discrete blocks, although it is recognized that such programs and components reside at various times in different storage components of the computing device 502, and are executed by the data processor(s) of the computer.

An implementation of the system and methods described herein may result in the storage or transmission of data, instructions, or other information across some form of computer readable media. Computer readable media can be any available media that can be accessed by a computer. By way of example, and not limitation, computer readable media may comprise "computer storage media" and "communications media." "Computer storage media" include volatile and non-volatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules, or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage

